producing a plurality of electrical signals representing the automaton B, the automaton B being equivalent to the automaton A without ε-transitions, the producing further comprising:

adding to E[p] non-empty-string transitions leaving each state "q" from the set of states reachable from "p" via a path labeled with an ~~ε-transitions~~ ε-transition, wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to "q" in the automaton "A" to produce the automaton ~~"~~B~~"~~ ~~equivalent to automaton A without ε-transitions~~.

20. (Currently Amended) A method of producing an equivalent weighted automaton ~~"~~B~~"~~ with no ε-transitions for any input weighted automaton ~~"~~A~~"~~ having a set of transitions "e", at least one of which is an ε-transition, a set of states "p", and a set of states "q", the method comprising:

inputting a plurality of electrical signals representing the weighted automaton A, the weighted automaton A further representing a plurality of hypotheses with associated weights; and

producing a plurality of electrical signals representing the automaton B with no ε-transitions, the producing comprising:

computing an ε-closure C[p] for each state "p" of the input weighted automaton "A";

for each of the states ~~state~~ "p", determining ~~the~~ non-ε-transitions from ~~state~~ the states "p";

for each of the states ~~state~~ "q" having a weight "w" within the computed ε-closure C[p]:

adding to outgoing transitions from the states "p", E[p], the non-ε-transitions leaving each of the states ~~state~~ "q"; and

if ~~state~~ one of the states "q" is part of a set of final states F, and if a corresponding one of the states ~~state~~ "p" is not part of the set of final states F:

defining the corresponding one of the states ~~state~~ "p" as included within the set of final states "F" and ~~the~~ a final weight ρ[p] as pre-⊗-multiplied by w, the ε-distance from state "p" to state "q" in the automaton A to produce the automaton B.

21. (Currently Amended) A method of removing string terms "a" from an automaton A having a set of states "p", a set of states "q", and a set of outgoing transitions from the set of states "p", E[p], the method comprising:

inputting a plurality of electrical signals representing the automaton A, the automaton A further representing a plurality of hypotheses with associated weights;

producing a plurality of electrical signals representing an automaton B from the automaton A, the producing comprising:

computing an a-closure for each state "p" of the automaton A; and

modifying E[p] by:

removing each transition labeled with a string term "a"; and

adding to E[p] a non-"a"-string transition, wherein each state "q" is left with its weights pre-⊗-multiplied by an a-distance from state "p" to a state "q" in the automaton A to produce the automaton B.

22. (Original) The method of claim 21, further comprising:

removing inaccessible states using a depth-first search of the automaton A.

23. (Currently Amended) The method of claim 21, wherein adding to E[p] a non-"a"-string ~~transitions~~ transition further comprises leaving ~~q~~ the state "q" with weights $(d[p,q] \otimes \rho[q])$ to E[p].

24. (Currently Amended) The method of claim 21, wherein the ~~step of~~ computing of an a-closure for each ~~input~~ state "p" of an input automaton A further comprises:

removing all transitions not labeled with a string "a" from automaton A to produce an automaton A_a;

decomposing A_a into its strongly connected components; and

computing all-pairs shortest distances in each of the strongly connected components ~~component~~ visited in reverse topological order.

25. (Currently Amended) The method of claim 21, wherein the ~~step of~~ computing of an a-closure for each ~~input~~ state "p" of an input automaton A further comprises:

decomposing A_a into its strongly connected components;

performing a single-source shortest-distance algorithm according to the following pseudo code:

for each $p \in Q$

do $d[p] \leftarrow r[p] \leftarrow \bar{0}$

$d[s] \leftarrow r[s] \leftarrow \bar{1}$

$S \leftarrow \{s\}$

while $S \neq 0$

do $q \leftarrow$ head [S]

DEQUEUE (S)

$r \leftarrow r[q]$

$r[q] \leftarrow \bar{0}$

for each $e \in E[q]$

do if $d[n[e]] \neq d[n[e]] \oplus (r \otimes w[e])$

then $d[n[e]] \leftarrow d[n[e]] \oplus (r \otimes w[e])$

$r[n[e]] \leftarrow r[n[e]] \oplus (r \otimes w[e])$

if $n[e] \notin S$

then ENQUEUE (S, n[e])

$d[s] \leftarrow \bar{1}$.

26. (Currently Amended) The method of claim 21, wherein the ~~step of~~ computing of the a-closure for each state "p" further comprises computing each of the a-closures according to the following equation:

$$C[p] = \{(q,w) : q \in a[p], d[p,q] = w \in K - \{\bar{0}\}\}.$$

27. (Currently Amended) The method of claim 26, wherein the ~~step of~~ modifying of ~~outgoing transitions of each state "p"~~ E[p] further comprises modifying the outgoing transitions of each state ~~p~~ "p" according to the following procedure:

~~(1)~~ for each $p \in Q$

~~(2)~~ do $E[p] \leftarrow \{e \in E[p] : i[e] \neq a\}$

~~(3)~~ for each $(q,w) \in C[p]$

~~(4)~~ do $E[p] \leftarrow E[p] \cup \{(p,a,w \otimes w',r) : (q,a,w',r) \in E[q], a \neq a\}$

~~(5)~~ if $q \in F$

~~(6)~~ then if $p \notin F$

~~(7)~~ then $F \leftarrow F \cup \{p\}$

~~(8)~~ $\rho[p] \leftarrow \rho[p] \oplus (\omega \otimes \rho[q])$.

28. (Currently Amended) The method of claim 27, wherein a state is a final state if some state "q" within a set of states reachable from "p" via a path labeled with an empty string is final and ~~the~~ a final weight is then: $\rho[p] = \bigoplus_{q \in C[p] \cap F} (d[p,q] \otimes \rho[q])$.

29. (Original) The method of claim 28, further comprising:

performing a depth-first search of the automaton A after removing the "a" strings.

30. (Currently Amended) A method of removing empty string terms from a transducer A having a set of states "p", a set of states "q", and a set of outgoing transitions from the set of states "p", E[p], the method comprising:

inputting a plurality of electrical signals representing the transducer A; and

generating a plurality of electrical signals representing a modified transducer A by:

computing an ε-closure for each ~~state~~ of the states "p" of the transducer A;

modifying E[p] by:

removing each transition labeled with an empty string; ~~and~~

adding to the E[p] a non-empty-string transition, wherein each ~~state~~ of the states "q" is left with its weights pre-multiplied by an ε-distance from ~~state~~ a corresponding one of the states "p" to a respective one of the states ~~state~~ "q" in the transducer A to generate the modified transducer A.

31. (Original) The method of claim 30, further comprising:

removing inaccessible states using a depth-first search of the transducer A.

32. (Currently Amended) The method of claim 30, wherein adding to E[p] non-empty-string transitions further comprises leaving <u>the states</u> q with weights (d[p,q] ⊗ ρ[q]) to E[p].

33. (Currently Amended) The method of claim 30, wherein the ~~step of~~ computing <u>of the</u> ε-closure for each ~~input state~~ <u>of the states "p"</u> of ~~an input~~ <u>the</u> transducer A further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each ~~component~~ <u>of the strongly connected components</u> visited in reverse topological order.

34. (Currently Amended) The method of claim 30, wherein the ~~step of~~ computing <u>of the</u> ε-closure for each ~~input state of an input~~ <u>of the states "p" of the</u> transducer A further comprises:

decomposing A_ε into its strongly connected components;

performing a single-source shortest-distance algorithm according to the following pseudo code:

for each $p \in Q$

do $d[p] \leftarrow r[p] \leftarrow \bar{0}$

$d[s] \leftarrow r[s] \leftarrow \bar{1}$

$S \leftarrow \{s\}$

while $S \neq 0$

do $q \leftarrow$ head [S]

DEQUEUE (S)

$r \leftarrow r[q]$

$r[q] \leftarrow \bar{0}$

for each $e \in E[q]$

do if $d[n[e]] \neq d[n[e]] \oplus (r \otimes w[e])$

then $d[n[e]] \leftarrow d[n[e]] \oplus (r \otimes w[e])$

$r[n[e]] \leftarrow r[n[e]] \oplus (r \otimes w[e])$

if $n[e] \notin S$

then ENQUEUE $(S, n[e])$

$d[s] \leftarrow \bar{1}$.

35. (Currently Amended) The method of claim 30, wherein the ~~step of~~ computing of the ε-closure for each ~~state~~ of the states "p" further comprises computing each the ε-closure according to the following equation:

$$C[p] = \{(q,w) : q \in \varepsilon[p], d[p,q] = w \in K - \{\bar{0}\}\}.$$

36. (Currently Amended) The method of claim 35, wherein the ~~step of~~ modifying of the outgoing transitions of each ~~state~~ of the states "p" further comprises modifying the outgoing transitions of each ~~state p~~ of the states "p" according to the following procedure:

~~(1)~~ for each $p \in Q$

~~(2)~~ do $E[p] \leftarrow \{e \in E[p] : i[e] \neq \varepsilon\}$

~~(3)~~ for each $(q,w) \in C[p]$

~~(4)~~ do $E[p] \leftarrow E[p] \cup \{(p, a, w \otimes w', r) : (q, a, w', r) \in E[q], a \neq \varepsilon\}$

~~(5)~~ if $q \in F$

~~(6)~~ then if $p \notin F$

~~(7)~~ then $F \leftarrow F \cup \{p\}$

~~(8)~~ $\rho[p] \leftarrow \rho[p] \oplus (\omega \otimes \rho[q])$.

37. (Currently Amended) The method of claim 36, wherein a state is a final state if some state "q" within a set of states reachable from <u>a corresponding state</u> "p" via a path labeled with an empty string is final and the final weight is then: $\rho[p] = \bigoplus_{q \in \epsilon[p] \cap F} (d[p, q] \otimes \rho[q])$.

38. (Original) The method of claim 37, further comprising:

performing a depth-first search of the transducer A after removing the empty strings.